Building a Dedicated Robotic Soccer System

Sorin Achim sorin@cs.cmu.edu
Peter Stone pstone@cs.cmu.edu
Manuela Veloso veloso@cs.cmu.edu

Computer Science Department
Carnegie Mellon University
Pittsburgh PA 15213-3890

Abstract

Robotic Soccer involves multiple agents that need to collaborate in an adversarial environment to achieve specific objectives. We have been building an architecture that addresses this integration of high-level and low-level reasoning as a combined system of mini-robots, a camera for perception, a centralized interface computer, and several client servers as the minds of the mini-robot players. In this paper we focus on the hardware design of our mini-robots. Our main purpose is to provide a detailed description of our design decisions so that others may learn from and replicate our efforts. Communication between the interface computer and the mini-robots is achieved through coded infrared radiation. The mini-robots can turn on the spot and move forward and backward with variable speed. Our design also allows a well-balanced use of the on-board power and current supplies.

1 Introduction

As robots become more adept at operating in the real world, the high-level issues of collaborative and adversarial planning and learning in real-time situations are becoming more important. An interesting emerging domain that is particularly appropriate for studying these issues is Robotic Soccer. Although realistic simulation environments exist [Noda, 1995; Sahota, 1993] and are useful, it is important to have some actual physical agents in order to address the full complexity of the task.

In this paper we discuss the design decisions we faced along with the impasses we encountered during our quest to build small robots, which call mini-robots, capable of playing Robotic Soccer. We are not the first to build robots for this domain [Asada *et al.*, 1994a; Sahota *et al.*, 1995], however our system is distinctly different from the others about which we know. Furthermore, with at least two Robotic Soccer competitions planned during the next two years [Stone *et al.*, 1996; Kitano *et al.*, 1997], there will surely be several more systems built in the near future. The purpose of this paper is to describe our current mini-robot system in as much detail as possible, so that others may benefit from both our setbacks and our successes. We aim for this paper to render our efforts as replicable as possible.

Although we will describe certain dead-ends we traversed, most of this paper is devoted to describing the current solution, rather than the path by which we came to it. We try to lay out the choices we faced at every step, but the worse decisions will mostly be inferable as the choices which did not lead to the current system. After a good deal of effort, we now have a working system with which up to sixteen cars can be independently controlled. Although our system is still evolving, the current version of our mini-robot system is fully implemented and functional. We find it important to present what we have learned to the Intelligent Robots community at this time.

Our current mini-robotic system is certainly usable for tasks other than Robotic Soccer, but since our main purpose in building the system was to work in the Robotic Soccer domain, we made most of our design decisions with this domain primarily in mind.

Robotic Soccer is an exciting domain for Intelligent Robotics for many reasons. The fast-paced nature of the domain necessitates real-time sensing coupled with quick behaving and decision making. Furthermore, the behaviors and decision making processes can range from the most simple reactive behaviors, such as moving directly towards the ball, to arbitrarily complex reasoning procedures that take into account the actions and perceived strategies of teammates and opponents. Opportunities, and indeed demands, for innovative and novel techniques abound.

A ground-breaking system for Robotic Soccer, and the one that served as the inspiration and basis for our work, is the Dynamo System developed at the University of British Columbia [Sahota *et al.*, 1995]. This system was designed to be capable of supporting several robots per team, but most work has been done in a 1 vs. 1 scenario. Sahota used this system to introduce a decision making strategy called reactive deliberation which was used to choose from among seven hard-wired behaviors [Sahota, 1994]. Subsequently, Ford used Reinforcement Learning (RL) techniques to choose from among the same hard-wired behaviors [Ford *et al.*, 1994]. Our system differs from the Dynamo system in several ways, most notably our use of infrared (IR) rather than radio waves for communication between the controlling computer and the cars. We also hope to do minimal hard-wiring, instead learning behaviors from the bottom up.

The Robotic Soccer system being developed in Asada's lab is very different from both the Dynamo system and from our own [Asada *et al.*, 1994a; 1994b]. Asada's robots are larger and are equipped with on-board sensing capabilities. They have been used to develop some low-level behaviors such as shooting and avoiding as well as a RL technique for combining behaviors [Asada *et al.*, 1994a; 1994b]. While the goals of this research are very similar to our own, the approach is different. Asada has developed a very nice sophisticated robot system with many advanced capabilities, while we have chosen to focus on producing a simple, robust design that will enable us to concentrate our efforts on learning low-level behaviors and high-level strategies. We believe that both approaches are valuable for advancing the state of the art of Robotic Soccer research.

One of the advantages of the Robotic Soccer domain is that it enables the direct comparison of different systems: they can be matched against each other in competitive tournaments. Systems like Dynamo and Asada's as well as ours and probably many others will come together at least twice in the next two years. In November 1996, there will be a Micro-Robot competition in Taejon, Korea called MIROSOT96. The call for participation is a good example of one possible set of precise specifications for this domain [Stone *et al.*, 1996]. Planning is also in progress for the 1997 robot soccer competition at IJCAI, to be called RoboCup97 [Kitano *et al.*, 1997].

Along with the real robot competition, RoboCup97 will also include a simulator-based tournament using the Soccer Server system designed by Noda [Noda, 1995]. While we continue working on our real-world system, we have been concurrently developing learning techniques in simulation [Stone and Veloso, 1995; 1996]. We eventually hope to transfer these learning techniques to the real system as we develop a complete Robotic Soccer architecture.

The rest of the paper is organized as follows. Section 2 gives an overview of the architecture of the entire Robotic Soccer system. Section 3 gives the detailed description of the existing cars which we view to be the main contribution of this paper. Section 4 draws conclusions from the paper and presents our on-going research agenda.

2 Overview of the architecture

Robotic Soccer consists in its essence of multiple agents that need to collaborate in an adversarial environment to achieve specific objectives. This requirement sets the need to have high-level decision making reasoning in addition to low-level behavioral procedures for the agents. The architecture of our system addresses this combination of high-level and low-level reasoning by viewing the overall system as the combination of the mini-robots, a camera over-looking the playing field connected to a centralized interface computer, and several clients as the minds of the mini-robot players. Figure 1 sketches the building blocks of the architecture.

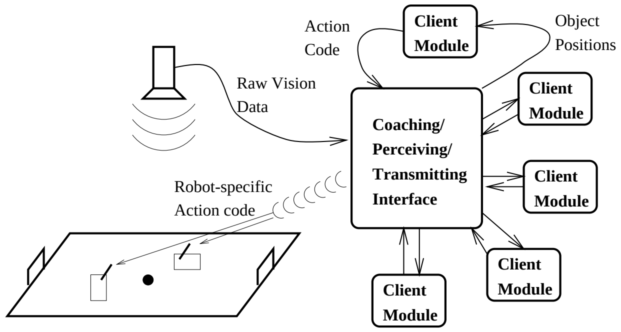

Figure 1: Our Robotic Soccer Architecture as a Distributed Deliberation and Reacting System.

In the rest of the paper we will describe in detail the specification of the mini-robots. As opposed to using radio communication as in a variety of other systems including [Sahota *et al.*, 1995], we use infrared communication for its multiple advantages, as identified in [Suzuki *et al.*, 1995].

Each mini-robot has on-board an infrared detector which allows it to receive and decode commands sent by a transmitter at the main host computer. The positioning of the mini-robots and of the ball will be perceived by a vision camera which passes this information to the main host computer. This processor maintains a map of the location of the mini-robots with different levels of confidence in its perception of the actual positions depending on the recency of the information gathered. The main host computer can be seen as the perception/action interface between the mini-robots and the high-level reasoning off-board clients: it perceives the environment, receives commands from the reasoning clients, and transmits these commands to the mini-robots. Commands can be broadcast or sent directly to individual agents. Each robot has a self identification binary code that is used in the infrared communication. Commands include positioning requests and navigation primitives, such as forward, backward, and turning moves at specific speeds. Robots can respond to positioning requests by turning on a high-intensity lighting source.

As we discuss in the future work section, we may provide the mini-robots with transmitting ability for direct communication between agents and some local sensing capabilities. We will consider these alternatives based on the empirical experience that we will be gathering in the near future using the vision camera.

In a nutshell, our architecture implements the overall robotic soccer system as a set of different platforms with different processing features. The mini-robots perform the physical navigation actions and can respond to positioning requests. Off-board computers perceive the environment through a vision camera, perform the high-level decision making and send commands to the mini-robots. Communication is done by infrared radiation.

3 Designing a team of moving agents remotely-controlled by a computer

In designing a team of moving agents remotely-controlled by a computer, some factors need to be considered. Depending on the task required to be done by the system, the technical parameters are set considering the most profitable implementations. In the particular case of a robotic soccer environment for example, the moving agent's dimensions are established relative to the playing field size. In the current specification, an area of 2x2m was defined. These dimensions were used in choosing the dimensions of the robots (approximately $7.5 \times 7.5 \times 10$ cm).

As the robots don't carry their own "brains" but are controlled from the computer, a very important part is the transmission link from the computer to the robots (Figure 2). As can be seen from Figure 2, the on-board system consists of motors, motor controllers, a decoder, and a receiver as well as batteries. The batteries are significant in that they comprise most of the weight of the on-board system. The off-board system consists of an IR transmitter, a coder, and a micro-controller. The

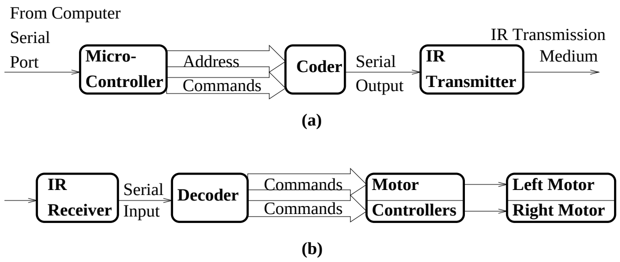

Figure 2: The transmission link connects the **(a)** off-board components and the **(b)** on-board components of the system.

transmission medium is IR radiation. This provides a robust and reliable connection that is free from interference. The emitter uses a carrier frequency of 40kHz ensuring very good noise rejection at the receiver. The information sent by the computer is first interpreted by a micro-controller (see Figure 2(a)). This interpretation is done by a C-program which converts the computer serial output to binary code. The commands for the robot motors are translated into 8-bit words while the robot addresses are transformed into 10-bit words. At the coder, the information bits are framed by synchronization blocks in order to allow for serial transmission. The framed control and address bit configuration is used for modulating the carrier IR frequency. The 40kHz-IR receiver on-board each robot filters and demodulates the signal and transmits it to the decoder (see Figure 3(b)). The decoder first examines the signal for the address bits and proceeds to analyze the command bits only in the case of a local address match.

A first attempt to get the system working was made using an RF AM transmitting-receiving system with the carrier frequency of 300 MHz. The transmitting and receiving modules were very small, light and compact, therefore suitable to be implemented in the small agents of the robotic system. Even though the modules worked well off the system, they failed to supply accurate data transmission when used in normal operating conditions. The main cause was the very noise caused by the brushed DC motors under the normal load. Its level was high enough to alter the data transmission in such a way that the decoders were not able to recognize any valid addresses. Thus the stopped robots always started to execute the first command but, after the motors were running, the noise altered the data and no robot was getting another command. Because the filtering methods being used did not work properly in this case (an amplitude modulated high frequency carrier and relatively noisy brushed DC motors) a simple IR solution was considered and implemented without the inconvenience caused by the RF noise.

3.1 The On-board System

First we describe in detail and in a bottom-up manner the on-board system. The most crucial part of a moving agent in this situation is the power source which should be able to supply enough current to move the robot and but must be as light as possible to allow the required autonomy between chargings. Since the batteries are used up quickly, cost effectiveness is also an important consideration. The battery that we have found to provide the best trade-off and which we are using is a rechargeable lead acid type, supplying 0.5 Amperes at 12 V.

The rest of the on-board system consists of two motors, motor controllers and, a decoder, and a receiver (see Figure 2(b)).

The Motors

In order to be able to move over the whole playing field, each robot has to be able to move straight back and forth, and to turn right and left. The right and left turns have to be made quickly and on the spot. Therefore, each robot carries two motors. Each motor drives a pair of wheels on one side of the robot. The width of the turn is controlled by changes applied to the motor on the side of the turn, i.e. by slowing, stopping or reversing it. The permanent magnet motors are reversible. The reversibility also allows the change in the movement direction to be made simply by reversing the polarity of the current in the motor windings.

The motors in the current design are brushed type, reversible DC motors with permanent magnets instead of field windings. Their operating voltage is rated between 2.5 and 12 V at a current draw of maximum 0.9 Amperes.

The Motor Controllers

To drive and control current in these motors an all DMOS full H-bridge with clamp diodes is used, along with an amplifier for sensing the load current, a comparator, a monostable , and a digital-to-analog converter (DAC) for the digital control of the chopping threshold. Together, they implement a fixed off-time chopper amplifier (Figure 3). Also incorporated are logic level shifting and drive blocks for digital control of the direction of the load current and for braking.

The H-bridge (Figure 4) consists of four DMOS power switches and associated clamp diodes connected in an H

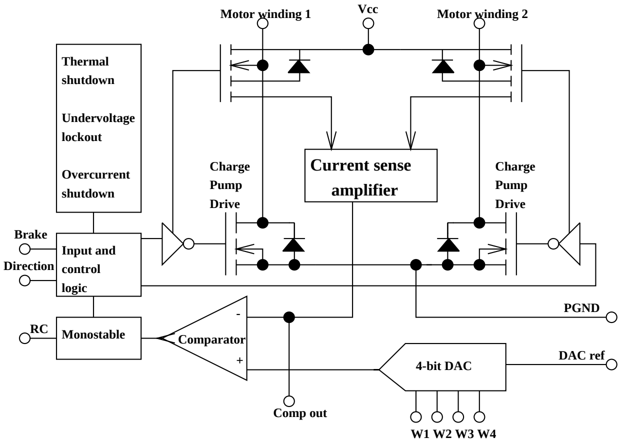

Figure 3: The fixed off-time chopper amplifier used by the motor controllers.

configuration. Turning ON a source switch and a sink switch in opposite halves of the bridge forces the full supply voltage—less the switch drops—across the motor winding. While the bridge remains in this state, the winding current increases exponentially towards a limit dictated by the supply voltage and the switch drops. Subsequently, turning OFF the sink switch causes a voltage transient that biases the diode of the other source switch. The diode clamps the transient at one diode drop above the supply voltage and provides an alternative current path. While the bridge remains in this state, it essentially shorts the winding and the winding current recirculates and decays exponentially towards zero. The above sequence repeats to provide a current chopping action that limits the winding current to the threshold. Chopping only occurs if the winding current reaches the threshold. During a change in the direction of the winding current, both the switches and the diodes provide a decay path for the initial winding current.

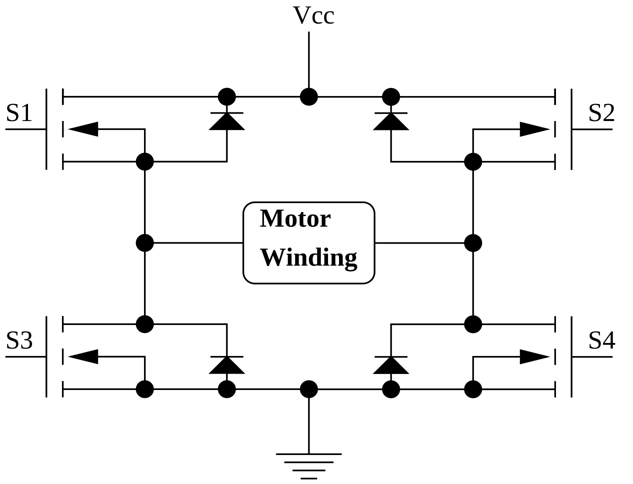

Figure 4: The H-bridge as part of the motors' current control.

The current sense amplifier (CSA) (Figure 5), another component of the micro-controller, uses one of the 4000 transistor cells of both upper power switches to provide a means for sensing the load current. It forces the voltage at the source of the sense device to equal that at the source of the power device; thus, the devices share the total drain current in proportion to the 1:4000 cell ratio. Only the current flowing from drain to source, the forward current, is registered at the output of the CSA. In these conditions, the CSA will provide around 250 μA per Ampere of total forward current conducted by the upper two switches of the power bridge, developing a potential across R13 that is proportional to the load current. In this way, modifying the R13 value will modify the gain of the chopper amplifier.

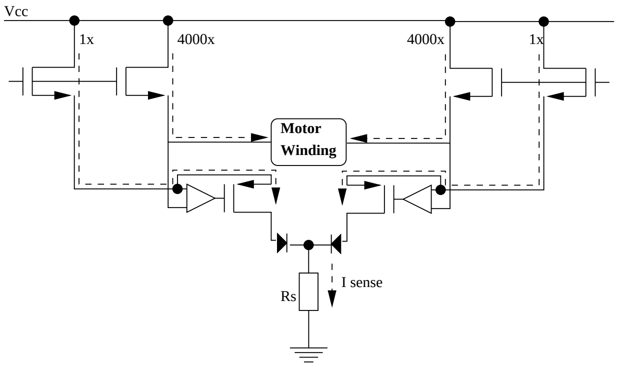

Figure 5: The current sense amplifier (CSA).

The DAC sets the threshold voltage for chopping at Vref $\times$ D/16, where D is the decimal equivalent (0 - 15) of the binary number applied at the four digital inputs of the DAC. The 5 V voltage reference is supplied by a 7805 voltage regulator IC. As the voltage at the CSA output surpasses that at the output of the DAC, the comparator triggers the monostable, and the monostable, once triggered, provides a timing pulse to the control logic. During the timing pulse, the power bridge shorts the motor winding, causing current in the winding to recirculate and decay slowly to zero. A parallel resistor-capacitor network connected to ground sets the timing pulse (the off-time) at about 1.1RC seconds.

In order to implement this motor-controller circuit an LMD 18245 (National Semiconductor) integrated circuit is used for each motor. Because we use 8 bits of data to control a robot, only 2 of the DAC's 4 digital inputs are used to obtain 3 speeds and the STOP state. Another bit sets the direction (high-forward and low-backward) and the remaining bits (1 for each motor) are used for special purposes if needed.

Step changes in current drawn from the power supply occur repeatedly during normal operation and may cause large voltage spikes in the power supply line. The voltage spikes need to be limited to less than the absolute maximum rating supported by the circuit. On the other hand, the initial load current tends to raise the voltage at the power supply rail at a change in the direction of the load current and the current transients caused by the reverse recovery of the clamp diodes tend to pull down the voltage at the power supply rail. Therefore, bypassing the power supply line at Vcc is required to protect the device and minimize the adverse effects of normal operation of the DC motor. Using both a 1μF high frequency ceramic capacitor and a 330μF aluminum electrolytic capacitor eliminates the problem. They have to be placed within one half inch of Vcc and their leads have to be as short as possible.

The Decoder

In the on-board circuitry, the motor controllers receive their commands from the decoder (see Figure /ref-fig1(b)). The control bits at the LMD 18245 input pins are supplied by the decoding circuitry (Figure 6) only after the decoder compares the serial input information twice with its local address and no error or unmatched codes are encountered. Simultaneously, a valid transmis-

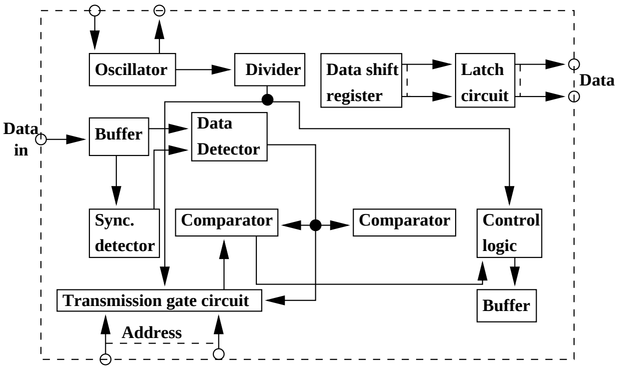

Figure 6: The decoding circuitry.

sion indicator will go high allowing a LED to be activated on the agent's board. The latch type data outputs follow the coder during a valid transmission and are latched in this state until the next valid transmission. All the decoder circuitry (buffers, comparators, oscillator, data detector, data shift register, latch circuit, transmission gate circuit, control logic, divider and synchronization detector) is implemented by a HT648L (Holtek) CMOS LSI integrated circuit. The low power, high noise immunity CMOS technology along with a low stand-by current and an operating voltage range of 2.4 to 12 V makes it a very good choice for remote control systems. Its built-in oscillator needs only a 5% resistor to set up the desired frequency. In our design, a 100 kHz oscillator frequency is selected by a 330 kOhm resistor. The oscillator is disabled in the stand-by state and activated as long as a logic high signal is applied to the data input. Even though the decoder is capable of validating up to 10 bit addresses, only 4 bits are used to address up to 16 robots. The extra bits enable future expandability of the system. The decoder's activity flowchart is shown in Figure 7.

The IR Receiver

The decoder's serial input information is supplied by the IR receiver after the received signal is amplified, filtered and demodulated. The receiver processing circuitry (Figure 8) consists of an amplifier, limiter, band pass filter, demodulator, integrator, and comparator. It is contained in the compact sized GP1U52X module (Archer), a hybrid IC/infrared detector circuit. Its operating voltage starts from 5 V, therefore allowing direct

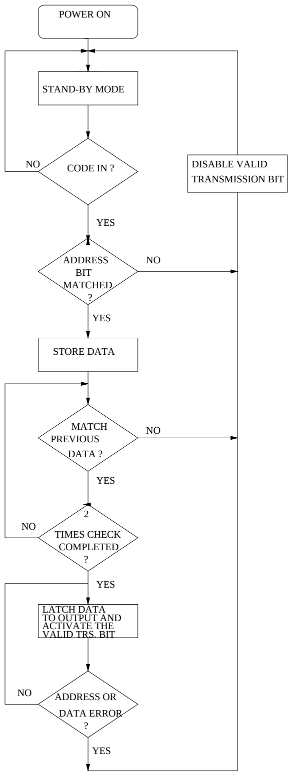

Figure 7: The decoder activity flowchart.

connection to TTL or CMOS components. The coil-free design provides total immunity from external noise induced by magnetic fields and the built-in low-pass filter on the power supply helps isolate the circuitry from power supply noise. The detector uses a pin photo diode that has its peak sensitivity in the near infrared range. The built-in filter blocks visible light to reduce or eliminate false operation caused by other light sources. The IR passband is 980 nm +/- 100 nm and the bandwidth (-3dB, 40 kHz) is 4 kHz. The output of the photo diode feeds into a preamplifier/limiter to provide a clean signal to the rest of the circuit. The band pass filter then rejects all signals outside the pass band and the resulting signal is fed to the demodulator, integrator and waveshaper circuit. The output is a clean waveform without the carrier. The interface circuit between the GP1U52X module and the decoder's input consists of 2 general purpose transistors in an amplifying configuration.

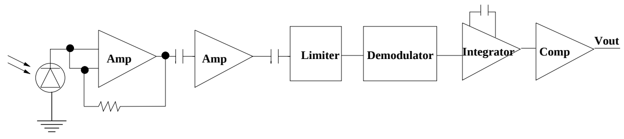

Figure 8: The IR receiver.

3.2 The Off-board system

The IR receiver gets its signal from the IR transmitter. This communication is the crucial link between the on-board system and the off-board system. Aside from the transmitter, the off-board system consists of the coder and the micro-controller (see Figure 2(a)).

The IR Transmitter

The IR transmitter is built around a 555 timer which is set to oscillate at 40 kHz. The signal from the coder's serial output modulates this oscillator and the resulting signal is applied to the IR LED array. The IR wavelength is 940 nm.

The Coder

The coder (Figure 9) scans two of the micro-controller's output ports (namely Port B and Port C as shown later) for the address and data bits and sends to the serial output a frame containing these bits packed with synchronization bit blocks. All the circuits required to

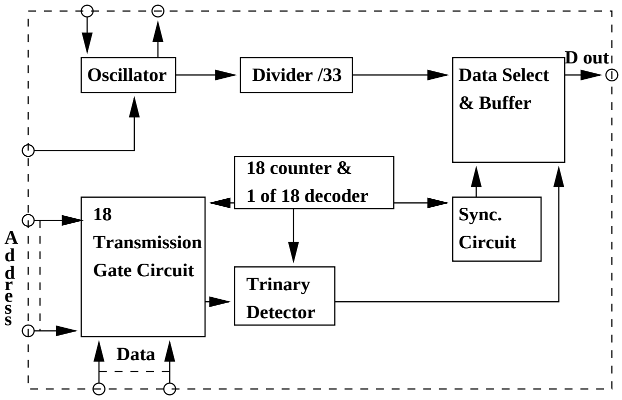

Figure 9: The coder.

perform this task (oscillator, divider, transmission gate circuit, counter, trinary detector, data selector, buffer and synchronization circuit) are contained in the HT648 (Holtek) CMOS integrated circuit. It has the same features as the matching decoder HT640 regarding power consumption, operating voltage and noise immunity. Its built-in oscillator is set to perform at the same frequency as the decoder's one in order to get the synchronization required by a valid data transmission. Upon receipt of a transmission enable command, the coder begins a 3 word transmission cycle. This cycle is repeated as long as the transmission enable input is held high. After the transmission enable falls low, the coder output completes its final cycle and then stops as shown in Figure 10.

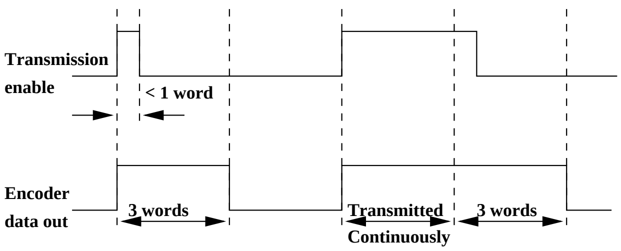

Figure 10: The coder transmission cycle.

An information word is composed of 4 periods as shown in Figure 11 and the coder activity flowchart is presented in Figure 12.

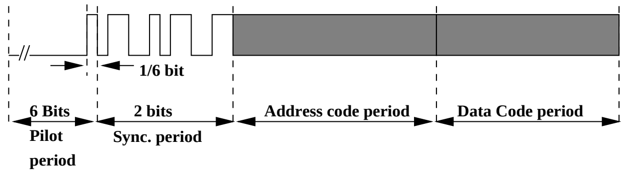

Figure 11: An information word is composed of 4 periods.

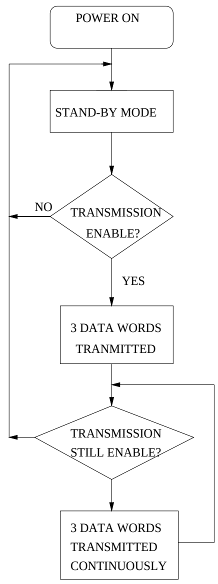

Figure 12: The coder activity flowchart.

The Micro-controller

The circuit used to receive the command and address codes sent by the computer through the serial port is based on a HCMOS MC68HC11A8 (Motorola) micro-controller having on-chip peripheral capabilities and a bus speed of 2 MHz. It includes a monitor/debugging program, one-line assembler/disassembler, host computer downloading capability, I/O expansion circuitry and RS-232C compatible I/O ports (Figure 13). This circuitry provides a very flexible and powerful interface between the computer and the coding/transmitting circuitry.

The micro-controller uses an 8kB ROM, 512 bytes EEPROM and 256 bytes of RAM and can address a supplementary 16k external RAM. Based on the enhanced M6800/M6801 instruction set, it allows a C-program residing in the external RAM to interpret the command and address codes. The C-program was edited and compiled by a cross-assembler on an IBM compatible computer and then uploaded in Motorola S-record format in the micro-controller's external RAM. The corresponding binary equivalents of the commands and addresses are directed to the micro-controller's output ports B and C. One of the 8 bits of the address port is used as a transmission enable bit. Three other bits are set to low logic level and only the remaining 4 bits are used for addressing. Half of the data port 8 bits are used to control a side motor and the remaining 4 bits will control the

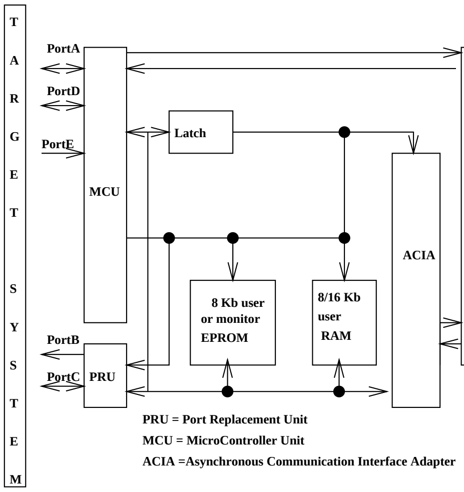

Figure 13: The micro-controller which provides the interface between the computer and the coding/transmitting circuitry.

other side motor. This design allows the computer to use its resources to perform the much more important task of "thinking" about the robots' movements and the strategy to be used to win the game.

In summary, we would like to highlight the main features of our system. Our IR-controlled mini-robots have several features, including:

- Turning quickly and on the spot;
- Forward and backward motion;
- Variable speed;
- Long range reception field;
- Position signalling through an LED;
- IR-coded communication with the interface;
- Well-balanced energy consumption.

Figure 14 shows a picture of a few of our current actual implemented mini-robots.

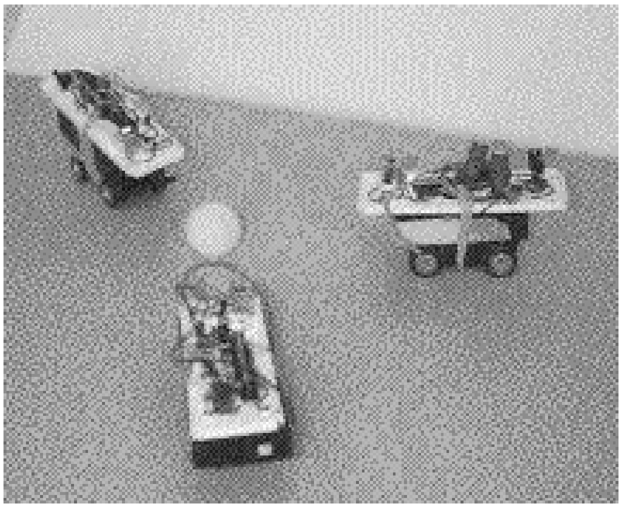

Figure 14: The actual mini-robots.

4 Conclusion

In this paper we provided a detailed description of our hardware implementation of mini-robots to play Robotic Soccer. Although we considered using off-the-shelf general-purpose robots, we found it to be less expensive to build our own. Furthermore we were able to include the exact features that we needed for our approach to Robotic Soccer.

As reported in this paper, we successfully achieved the implementation of the mini-robot players. We consider this a major step towards the development of the overall system. We are now ready to move on to the next phase. Therefore, our on-going and future work includes the incorporation of the vision system, the perception-based control of the individual agents, and the high-level strategy and behavior learning for the real robotic system. For the perception-based control, we will build upon and extend multiple robot path planning which assumes global control [Ferrari *et al.*, 1995]. Our added challenge is that each mini-robot is controlled by a separate client. We may eventually include limited on-board sensing and intra-robot communication as in [Suzuki *et al.*, 1995] to aid in path planning and obstacle avoidance. Since we have been working on learning issues in a simulated environment [Stone and Veloso, 1995; 1996], we expect to use similar techniques in the real robotic system.

We plan to complete the system and participate at MIROSOT96 and RoboCup97. We hope that our research and this detailed description of our design will encourage others to participate as well.

Acknowledgements

We would like to acknowledge Michael Fung, Sam Miller, and Michael Collins for their help in initial phases of the project.

This research is sponsored in part by the Wright Laboratory, Aeronautical Systems Center, Air Force Materiel Command, USAF, and the Advanced Research Projects Agency (ARPA) under grant number F33615-93-1-1330. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Wright Laboratory or the U. S. Government.

References

[Asada *et al.*, 1994a] Minoru Asada, Shoichi Noda, Sukoya Tawaratsumida, and Koh Hosoda. Purposive behavior acquisition on a real robot by vision-based reinforcement learning. In *Proc. of MLC-COLT (Machine Learning Confernce and Computer Learning Theory) Workshop on Robot Learning*, pages 1–9, 1994.

[Asada *et al.*, 1994b] Minoru Asada, Eiji Uchibe, Shoichi Noda, Sukoya Tawaratsumida, and Koh Hosoda. Coordination of multiple behaviors acquired by vision-based reinforcement learning. In *Proc. of IEEE/RSJ/GI International Conference* *on Intelligent Robots and Systems 1994 (IROS '94)*, pages 917–924, 1994.

[Ferrari *et al.*, 1995] C. Ferrari, E. Pagello, J. Ota, and T. Arai. Planning multiple autonomous robots motion in space and time. In *Proceedings of the 1995 IEEE/RSJ Internation Conference on Intelligent Robots and Systems*, volume 2, pages 253–259. IEEE Computer Society Press, 1995.

[Ford *et al.*, 1994] Roger Ford, Craig Boutilier, and Keiji Kanazawa. Exploiting natural structure in reinforcement learning: Experience in robot soccer-playing, 1994. Unpublished Manuscript.

[Kitano *et al.*, 1997] Hiroaki Kitano, Yasuo Kuniyoshi, Itsuki Noda, Minoru Asada, Hitoshi Matsubara, and Eiichi Osawa. RoboCup: A challenge problem for AI. *AI Magazine*, 18(1):73–85, Spring 1997.

[Noda, 1995] Itsuki Noda. Soccer server : a simulator of RoboCup. In *Proceedings of AI symposium '95*, pages 29–34. Japanese Society for AI, December 1995.

[Sahota *et al.*, 1995] Michael K. Sahota, Alan K. Mackworth, Rod A. Barman, and Stewart J. Kingdon. Real-time control of soccer-playing robots using off-board vision: the dynamite testbed. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 3690–3663, 1995.

[Sahota, 1993] Michael K. Sahota. Real-time intelligent behaviour in dynamic environments: Soccer-playing robots. Master's thesis, University of British Columbia, 1993.

[Sahota, 1994] Michael K. Sahota. Reactive deliberation: An architecture for real-time intelligent control in dynamic environments. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, pages 1303–1308, 1994.

[Stone and Veloso, 1995] Peter Stone and Manuela Veloso. Broad learning from narrow training: A case study in robotic soccer. Technical Report CMU-CS-95-207, Computer Science Department, Carnegie Mellon University, 1995.

[Stone and Veloso, 1996] Peter Stone and Manuela Veloso. Beating a defender in robotic soccer: Memory-based learning of a continuous function. In David S. Touretzky, Michael C. Mozer, and Michael E. Hasselmo, editors, *Advances in Neural Information Processing Systems 8*, pages 896–902, Cambridge, MA, 1996. MIT press.

[Stone *et al.*, 1996] Peter Stone, Manuela Veloso, and Sorin Achim. Collaboration and learning in robotic soccer. In *Proceedings of the Micro-Robot World Cup Soccer Tournament*, Taejon, Korea, November 1996. IEEE Robotics and Automation Society.

[Suzuki *et al.*, 1995] Sho'ji Suzuki, Hajime Asama, Akira Uegaki, Shin'ya Kotosaka, Takanori Fujita, Skihiro Matsumoto, Hayato Kaetsu, and Isao Endo. An infra-red sensory system with local communication for cooperative multiple mobile robots. In *Proceedings of the 1995 IEEE/RSJ Internation Conference on Intelligent Robots and Systems*, volume 1, pages 220–225. IEEE Computer Society Press, 1995.